K-means

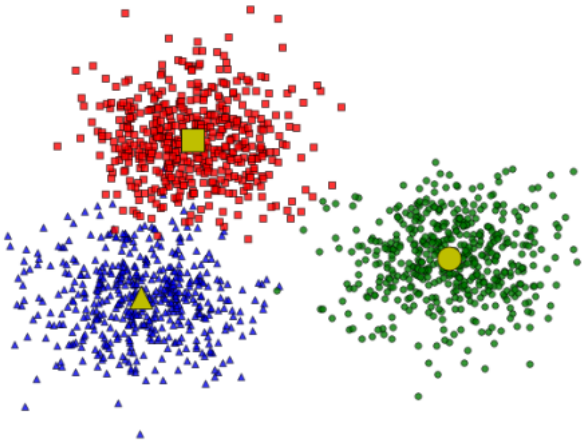
Hoàng Vũ Minh

K-means là một thuật toán Unsupervised Learning, tức là ta hoàn toàn không bết label của từng điểm trong dữ liệu, mục tiêu của ta là làm thế nào chia chúng về các cụm sao cho các điểm dữ liệu trong 1 cụm thì tương đồng/có tính chất gần giống nhau.

Khái niệm cụm: cluster (cụm) là tập hợp các điểm *ở gần nhau trong một không gian nào đó* (không gian này có thể có rất nhiều chiều trong trường hợp thông tin về một điểm dữ liệu là rất lớn)

1. Ý tưởng cơ bản:

Ta coi mỗi cluster có một điểm đại diện, ta có thể gọi nó là tâm cụm (*center*). Những điểm xung quanh mỗi center thuộc vào cùng nhóm với center đó. Như vậy cách dễ nhất: xem khoảng cách từ các điểm đến các tâm cụm, gần tâm cụm nào hơn thì thuộc vào cụm đó.



Vì vậy việc khó chính là đi tìm các tâm cụm, sau khi tìm các tâm cụm, bài toán trở nên đơn giản hơn rất nhiều. Mục đích cuối cùng của thuật toán phân nhóm này là: từ dữ liệu đầu vào và số lượng nhóm chúng ta muốn tìm, hãy chỉ ra center của mỗi nhóm và phân các điểm dữ liệu vào các nhóm tương ứng. Giả sử thêm rằng mỗi điểm dữ liệu chỉ thuộc vào đúng một nhóm.

2. *Hàm mất mát*

Dựa vào mô tả và mục tiêu bên trên, hám mất mát ta cần sẽ là một hàm cho ta khả năng đo tổng khoảng cách tự một điểm đến tâm cụm của nó. Như vậy nếu giá trị hàm mất mát càng nhỏ thì các điểm càng gần với tâm cụm → càng tốt.

Hàm mất mát đơn giản chỉ là:

$$\mathcal{L} = \sum_{i=1}^{m} \sum_{k=1}^{K} 1\{c_i = k\} ||x_i - \mu_k||^2$$

Trong đó: c là một vector để xem điểm dữ liệu x nào thuộc về cụm nào.

- μ là K-mean vector, là vector trung bình của cụm thứ k.
- m là số mẫu
- K là số cụm
- nếu xi thuộc cụm k thì ci =k sẽ cho giá trị 1 và ngược lại là 0.

3. *Bài toán tối ưu*

Dựa vào hàm mất mát bên trên, bài toán tối ưu là tìm c và μ sao cho Loss nhỏ nhất. Vì không thể tìm cả 2 cùng lúc nên người ta thường thực hiện việc này bằng cách cố định 1 trong 2 rồi tìm cái còn lại tốt nhất.

- Fix μ tìm c
- Fix c tìm μ

Đây là tóm tắt quá trình hoạt động của thuật toán

1 Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly

2 Repeat until convergence:

For every i, set $c^i = \arg\min_j ||x^i - \mu_j||^2$

For each j, set $\mu_j = \frac{\sum_{i=1}^{m} 1[c^i = j] x^i}{\sum_{i=1}^{m} 1[c^i = j]}$

Công thức cập nhật tâm cụm μ chính là lý do vì sao thuật toán được gọi là K-means, đơn giản là vì:

- mẫu số chính là phép đếm *số lượng các điểm dữ liệu* trong cluster
- Còn tử số chính là *tổng các điểm dữ liệu* trong cluster

Vì đều phải tính lại tâm cụm mới sau mỗi lần lặp nên nó cũng khá linh hoạt nếu có thêm dữ liệu mới.

4. *Một vài nhược điểm của K-means:*

- Cần biết trước số cụm K
- Kết quả cuối cùng phụ thuộc khá nhiều vào việc khởi tạo tâm cụm. Nếu xui mọi thứ sẽ rất lâu và kết quả có thể không hoàn toàn tốt.
- Các cluster cần có dạng hình tròn/cầu, và kích thước cụm tương đương nhau: việc này liên quan đến việc tính toán khoảng cách Euclid. nhìn chung Kmeans sẽ không hoạt động tốt nếu bỏ qua điều kiện này.

5. *Làm thế nào để biết bao nhiêu cụm là tốt: Silhouette Score*

Silhouette Score là gì: Đây là một metric để đo lường xem mức độ phù hợp của một điểm với cụm hiện tại của nó, nhìn chung ta có thể hiểu sihou càng cao thì phân cụm càng tốt, số cụm đó là tốt.

Dưới đây là công thức sihou cho 1 điểm dlieu. Khi tính cả set thì tổng all vào:

$$s = \frac{b - a}{\max(a, b)}$$

Trong đó:

- a (intra-cluster distance): trung bình khoảng cách từ 1 điểm dlieu đến tất cả các điểm dlieu khác trong cụm của nó.
- b (inter-cluster distance): Khoảng cách giữa điểm dữ liệu và cụm gần nhất mà điểm đó không thuộc về.

Sau đó thường ngta dùng mean(sihou) tức là sẽ tính trung bình sihou của tất cả các điểm dữ liệu.

Giá trị của hệ số phân tích Silhouette nằm trong khoảng từ $[-1,1]$.

- Điểm số bằng 1 biểu thị tốt nhất, nghĩa là điểm dữ liệu i rất chặt chẽ trong cụm mà nó thuộc về và cách xa các cụm khác.
- Giá trị tệ nhất là -1.
- Các giá trị gần 0 biểu thị các cụm bị chồng lấn.